AMENDMENTS TO THE CLAIMS

This listing of claims replaces all prior versions, and listings, of claims in the application:

1-7. (canceled)

8. (new) A deterministic finite state automaton (FSA), responsive to an *n*-symbol alphabet, adapted to match, in parallel, *m* patterns each comprised of sequentially ordered symbols selected from said alphabet, the FSA comprising:

an initial state adapted to store therein *n* nextState transitions, each transition being responsive to a respective one of said symbols in said alphabet;

a plurality of states, each corresponding to one symbol contained at a given position in one or more of said *m* patterns, and being adapted to store therein *n* nextState transitions, each transition being responsive to a respective one of said symbols in said alphabet;

wherein:

said initial state has stored therein a plurality of transitions each to a first state responsive to a first symbol of one or more of said *m* patterns;

each state which corresponds to position *i* in one or more of said *m* patterns having stored therein *j* transitions to the *j* next sequential states each responsive to a particular symbol at position *i+1* in one or more of said *m* patterns, such that each of said *m* patterns is associated with one unique sequential set of successive transitions from the initial state;

if, after *s* sequential symbols in any one A of said *m* patterns, a sequence of no more than *t* sequential symbols corresponds to the first *t* symbols of a pattern B from amongst said *m* patterns, where *t* is at least one but less than the number of symbols in pattern B, then the (*s* + *t*) state of said unique sequential set associated with pattern A of successive transitions from the initial state also has

stored therein a transition to the (t + 1) state of said unique sequential set associated with pattern B of successive transitions from the initial state.

9. (new) The FSA of claim 8 wherein all transitions stored in each of said states, other than the transitions detailed in claim 8, comprise transitions to said initial state, each responsive to a respective one of said symbols of said alphabet.

10. (new) A method for creating the deterministic finite state automaton (FSA) of claim 9, the method comprising the steps of:

(1) creating a partial graph by:

(1.1) creating an initial state, and storing therein n nextState transitions to said initial state, each responsive to a respective one of said symbols in said alphabet;

(1.2) selecting said first pattern;

(1.3) for said selected pattern:

(1.3.1) selecting as a current state said initial state;

(1.3.2) selecting a first symbol in said selected pattern;

(1.3.3) selecting as a next state the nextState transition stored in said selected current state which is responsive to the selected symbol;

(1.3.4) if said next state is said initial state:

(1.3.4.1) creating a next state, and storing therein n nextState transitions to said initial state, each responsive to a respective one of said symbols in said alphabet; and

(1.3.4.2) storing in said selected current state a transition to said next state responsive to said selected symbol;

(1.3.5) selecting as said current state said selected next state;

(1.3.6) if said selected pattern comprises at least one additional symbol, selecting a next successive symbol in said selected pattern and returning to step (1.3.3); and

(1.3.7) if said selected pattern comprises said first pattern, selecting said second pattern and returning to step (1.3.1); and

(2) completing said partial graph by:

(2.1) selecting said first pattern;

(2.2) for said selected pattern:

(2.2.1) selecting a first symbol in said selected pattern;

(2.2.2) selecting as said current state the nextState transition stored in said initial state which is responsive to the selected symbol;

(2.2.3) if said selected pattern comprises at least one additional symbol:

(2.2.3.1) selecting a next successive symbol in said selected pattern;

(2.2.3.2) selecting as said current state the nextState transition stored in said selected current state which is responsive to the selected symbol;

(2.2.3.3) if a first-in-first-out queue has q states enqueued therein, where q is at least one:

(2.2.3.3.1) removing from said queue the first state enqueued therein;

(2.2.3.3.2) selecting as a temporary state said state removed from said queue;

(2.2.3.3.3) selecting as said temporary state the nextState transition stored in the temporary state selected in step (2.2.3.3.2) which is responsive to the selected symbol;

(2.2.3.3.4) if said selected temporary state is not said initial state:

(2.2.3.3.4.1) for each nextState transition in the selected current state which corresponds to the initial state, storing in said selected current state the corresponding nextState transition stored in said selected temporary state;

(2.2.3.3.4.2) enqueueing on said queue the selected temporary state; and

(2.2.3.3.4.3) if said queue has at least one of said q states enqueued therein, returning to step (2.2.3.3.1);

(2.2.3.4) selecting as said temporary state the nextState transition stored in said initial state which is responsive to the selected symbol;

(2.2.3.5) if said selected temporary state is not said initial state:

(2.2.3.5.1) enqueueing on said queue said selected temporary state; and

(2.2.3.5.2) for each nextState transition in the selected current state which corresponds to the initial state, storing in said selected current state the corresponding nextState transition stored in said selected temporary state;

(2.2.3.6) if said selected pattern comprises at least one additional symbol, selecting a next successive symbol in said selected pattern and returning to step (2.2.3.1); and

(2.3) if said selected pattern comprises said first pattern, selecting said second pattern and returning to step (2.2).

11. (new) The method of claim 10 wherein each of said states comprises an array of n elements, each element adapted to store a transition responsive to a respective one of said n symbols.

12. (new) The method of claim 10:

wherein each of said states is adapted to store an action to be performed upon transition to said state; and

wherein said step (1.3.5) is further characterized as:

(1.3.5a) selecting as said current state said selected next state; and

(1.3.5b) storing into said selected current state a predetermined action associated with said selected pattern.

13. (new) The method of claim 12 wherein each of said states comprises:

an array of *n* elements, each element adapted to store a transition responsive to a respective one of said *n* symbols; and

an action associated with said state.

14. (new) The method of claim 13 wherein each state having an action associated therewith is adapted to perform said action in response to transition thereto during a match operation.

15. (new) The method of claim 12:

wherein said step (2.2.3.5.2) is further characterized as:

(2.2.3.5.2a) for each nextState transition in the selected current state which corresponds to the initial state, storing in said selected current state the corresponding nextState transition stored in said selected temporary state; and

(2.2.3.5.2b) storing into said selected current state said predetermined action associated with said selected pattern.

16. (new) The method of claim 15 wherein each of said states comprises:

an array of *n* elements, each element adapted to store a transition responsive to a respective one of said *n* symbols; and

an action associated with said state.

17. (new) The method of claim 16 wherein each state having an action associated therewith is adapted to perform said action in response to transition thereto during a match operation.

18. (new) The FSA of claim 8 wherein:

each final state reached by each unique sequential set of successive transitions from the initial state associated with each of the *m* patterns has stored therein an action to be performed upon transition to said final state.

19. (new) A method of using the FSA of claim 18, the method comprising the steps of:

receiving an input stream of symbols, at least some of which are selected from said alphabet;

presenting to said FSA each of said received symbols; and

when a sequence of symbols in said input stream is found by said FSA to match one of the said *m* patterns, performing said action.